

INFORMATION ACCESS, COLLABORATION AND INTEGRATION SYSTEM AND METHOD

RELATED APPLICATION INFORMATION

5 This application claims the benefit of United States Provisional Application
Serial No. 60/182,398 entitled "Collaboration System Infrastructure For
Communication Between Computer System Users" filed on February 14, 2000.

FIELD OF THE INVENTION

10 The present invention relates generally to the field of computer systems and
programs, and in particular to a computer implemented collaboration system and
method that provides an infrastructure for access to multiple domain data sources, and
allows data from those sources to be analyzed and manipulated within a multi-user
distributed environment where all visualization, processing, and agent applications
15 work collaboratively.

BACKGROUND OF THE INVENTION

Referring to FIG. 1, older data access and analysis systems were generally
built as large application programs where most, if not all, system capabilities were
20 tightly coupled within the application. Having one large application can be difficult
and costly to maintain. Changes to a single capability within the application can have
a ripple effect throughout the source code requiring extensive changes to other areas
of the application. Repeated modification to the application can result in a system
that is so large and complex that enhancements become too cost prohibitive to
25 implement. Such large data access analysis systems have typically been built as a
result of analysis that failed to identify an existing infrastructure that met the needs of
future collaborative systems. To summarize, with present data access and analysis
systems, tools are generally restricted to a specific data source, there is difficulty in
analyzing data from various sources, the systems are costly to enhance, and there is an
30 inability to collaborate on multiple data sources at the same time to solve a problem.

SUMMARY OF THE INVENTION

Accordingly, one object of the present invention is to provide a collaborative
system with simplified access to multiple domain data sources.

Another object of the present invention is to provide a collaborative system with full lifecycle management.

A further object of the invention is to provide a collaborative system that allows manipulation of dissimilar data using common tools.

5 Yet another object of the present invention is to provide a collaborative system providing for the simultaneous manipulation of various data sources within the same tool.

10 A still further object of the present invention is to provide a collaborative system environment that provides a path for new systems to be developed by making maximum use of legacy system capabilities.

An additional more object of the present invention is to provide for new capabilities to be added to existing systems without requiring additional investment in the old system architectures.

15 Another additional object of the present invention is to provide a collaborative system with system scalability, extensibility, and maintainability within a collaborative framework exceeding that provided by existing technologies.

One more object of the present invention is to provide a collaborative system having an infrastructure that is domain independent.

20 Yet one more object of the present invention is to allow collaboration groups solving different problems to apply different visualization and control properties to their data collections than other collaboration groups.

SUB These and other objects and advantages are realized by various aspects of the present invention. According to one aspect of the present invention, a computer implemented collaboration system includes at least one repository server, at least one client tool, and at least one data channel server. The repository server is associated with at least one data source and is enabled for accessing data items within its associated data source(s) using access methods native to its associated data(s) source. The client tool is enabled for displaying the data items within said the data source(s) associated with the repository server on a user terminal connectable with the computer implemented collaboration system. The data channel server provides an interface between the repository server and the client tool. In this regard, the repository server, the client tool, and the data channel server may be implemented within a Common Object Request Broker Architecture (CORBA) framework. The computer implemented collaboration system also includes at least one extended property such

as, for example, a visualization property or a control property for use in displaying the data items with the client tool. The extended property is associated with each data item within the data source(s) and is maintained by the data channel server. In this regard, the extended property may be maintained in the data channel server in a directed a-cyclic graphical form.

The computer implemented collaboration system may further include a query viewing client tool and a library server. The query viewing client tool is enabled for use in constructing queries (e.g., standing or static queries) for selecting data from the data source(s) meeting particular criteria. The library server provides an interface between the query viewing client application and the repository server. The computer implemented collaboration system may also include a conference manager client tool enabled for managing communication between multiple user terminals connectable with the computer implemented collaboration system.

Sub 1 According to another aspect of the present invention, a computer implemented method for collaborative access to and manipulation of data items within at least one data source includes the step of creating a document representing data items the data source(s). In this regard, the document may be created by executing a static query of the data source(s) to identify data items meeting specified query criteria at a moment in time, or evaluating the query continually, either periodically or as new data arrives, providing a standing query of said the data source(s) to identify data items meeting specified query criteria.

After it is created, the document is associated with at least one conference that is accessible to at least one user terminal. In this regard, the document may be associated with the conference by using a pointing device to drag and drop a representation, displayed on the user terminal, of the document onto a representation, displayed on the user terminal, of said the conference.

The method also includes the steps of displaying the data items within the document on the user terminal user terminal and the step of maintaining an instance of at least one extended property associated with each data item represented in the document. The instance of the extended property is only available within the conference with which the document is associated. The extended property may, for example, comprise a visualization property (e.g., a preferred color or symbol) or a control property (e.g., a reporting task) for use in displaying the data items on the user terminal in the displaying step.

The present invention extends the scope of collaboration beyond simple whiteboard, audio chat, and video where all participants have exactly the same view of the data, from the same perspectives, identical picture, and scale. In the present invention, this mode is supported, but others are as well. Depending on viewer policy, participants have the ability to see the same data in the same presentation; have the ability to share scale and perspective, but allow different views sizes and locations; and have the ability to use different tools, different perspectives, and different scales while still working together on the same data.

These and other aspects and advantages of the present invention will be apparent upon review of the following Detailed Description when taken in conjunction with the accompanying figures.

DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention and further advantages thereof, reference is now made to the following Detailed Description, taken in conjunction with the drawings, in which:

FIG. 1 illustrates limitations of present collaborative systems;

FIG. 2 is a flow chart of the collaborative environment;

FIG. 3 is an overview of how clients participate with a document;

FIG. 4 is a flow chart illustrating how a client interacts with data on a document;

FIG. 5 illustrates the collaboration process on multiple views;

FIG. 6 is a flow chart illustrating flexibility and collaboration;

FIG. 7 is a pictorial summation of how a client accesses information;

FIG. 8 is a flow chart of the architectural strategy;

FIG. 9 is a flow chart of the services based architecture;

FIG. 10 is a flow chart of the system to extend the infrastructure;

FIG. 11 is a flow chart of minimum level integration with legacy systems;

FIG. 12 is a flow chart of mid-level integration with legacy systems;

FIG. 13 is a flow chart of full integration with legacy systems;

FIG. 14 is a chart summarizing the importance of having a data-centric collaboration network;

FIG. 15 is a first chart illustrating collaboration application management;

FIG. 16 is a second chart illustrating collaboration application management;

- FIG. 17 is a flow chart of the repository query and document management;
 FIG. 18 is a map and white-board interaction;
 FIG. 19 illustrates the function of the extended properties editor;
 FIG. 20 illustrates the output from an X-Y plotter;
 5 FIG. 21 illustrates the output in a list viewer;
 FIG. 22 illustrates the chat tool capability;
 FIG. 23 is a flow chart illustrating the performance metrics;
 FIG. 24 illustrates the high level interaction between various information
 access service (IAS) components in accordance with the USGIS;
 10 FIGS. 25-27 illustrate the lower level interaction between IAS components;
 FIGS. 28-32 illustrate the inheritance structure of the various IAS components
 illustrated in FIGS. 24-27 is illustrated in FIGS. 28-32;
 FIG. 33 illustrates the data channel services framework;
 FIG. 34 illustrates the versioning of data changes in the data channel;
 15 FIG. 35 illustrates the components that make up a "feature collection";
 FIGS. 36-37 illustrate the directed a-cyclic graph data structure format.

DETAILED DESCRIPTION

Referring generally to FIGS. 2-10, the computer implemented collaboration
 20 system 10 of the present invention incorporates a component-based infrastructure
 providing an architectural foundation for developing/incorporating advanced
 capabilities into new or legacy systems. The infrastructure incorporates a data centric
 approach where domain information is extended with control and visualization
 attributes and presented as self-describing objects. Data access is provided through
 25 industry standard interfaces adding to the ease of integration with legacy and
 commercial applications. The collaboration system builds on a data centric
 philosophy to provide key foundation frameworks for data access, collaboration, and
 component integration.

The collaboration system infrastructure is designed to integrate with existing
 30 collaborative products such as, for example, Net Meeting, Sun Forum, CVW,
 InfoWorkspace and Placeware, and to make available additional collaborative
 capabilities not provided by existing tools. Specifically, the collaboration system
 infrastructure provides access to multiple domain data sources, and allows data from
 those sources to be analyzed and manipulated within a multi-user distributed

environment where all visualization, processing, and agent applications work collaboratively.

5 The collaboration system is a fully distributed architecture allowing each service to be configured and executed anywhere within the network. It is built upon an architectural framework including CORBA and Java. The infrastructure is platform independent with demonstrated operation under heterogeneous operating environments consisting of Microsoft® Windows 9x, Windows NT, Windows 2000, and Unix (e.g., Solaris 2.x). The collaboration system is based on established and emerging government and commercial open standards including the Geospatial
10 Information Access Specification (GIAS), OpenGIS, and Document Object Model (DOM). All interfaces to the collaboration system infrastructure are provided through standard Interface Definition Language (IDL) ensuring adaptability to legacy systems written in Java, C, C++, Ada, or any other language with IDL bindings.

Still referring generally to FIGS. 2-10, the collaboration system data access
15 framework incorporates an adaptive repository layer that accesses the domain data through the access methods native to the data source. This enables data from any data source (real-time data feed, object data base, relational database, file system, etc.) to be accessed from the infrastructure. The repository approach is non-intrusive such that domain data sources do not need to be modified in any way. The repository acts
20 as a gateway to the native data. The repository is responsible for describing the data and making the data available through industry standard interfaces. This alleviates the need for client applications to have any knowledge of data location or specific access logic unique to the data source.

Extensibility and flexibility are key attributes of the collaboration system
25 infrastructure. Data is made available in a self-describing format such that client applications learn about the data and are able to manipulate the data without any a priori knowledge of its intrinsic structure. Client viewers are subsequently able to manipulate data from a variety of different domain sources without requiring any specialized software. Therefore, adding a new data source or changing the structure
30 of an existing data source requires no changes to the infrastructure or client applications. In addition, adding client applications that provide extended capabilities are able to manipulate data within any available data source.

Referring more specifically to FIG. 2, there is shown an overview of a collaborative interoperable context 100 that is provided by the computer implemented

collaboration system 10 of the present invention. Within the collaborative interoperable context 100 one or more conferences 102 are provided in which multiple participants 104 are able to collaboratively access and manipulate data from one or more data sources at the same time to solve a problem. The participants 104, who may be geographically remote from one another, access the conferences 102 via user terminals 106 connected to a data network 108. The participants 104 to a conference 102 are able to access and manipulate the data through one or more documents 110 that represent the data sources. For example, as is illustrated, within a conference 102 of a context 100 relating to an intelligence gathering and analysis operation there may be documents 110 representing logistics data, signal intelligence data, terrain data, map data, image data and the like, together providing a common operational picture. It will be appreciated that although the illustrated context 100 is of an intelligence related nature, the collaborative interoperable context 100 may relate to many other non-military related matters.

The context 100 provides a higher order organization for the conference 102. A context 100 may be a floor in a building, a region within a country or a conference room. Contexts 100 may be entered by participants 104 as a room would be entered and conferences 102 can be established. Conferences 102 provide the context 100 to drop documents 110 for collaboration. A document 110 dropped within a conference 102 will have an associated data channel that will maintain and make available the collection of information represented by the document 110 as well as any extended visualization or control properties.

Referring now to FIGS. 2-4, each document 110 represents data from a corresponding data source 112. A document 110 may be created by performing queries against the corresponding data source 112 or it may be created as an empty document 110 to be populated using interactive tools. In the former case, the query may be one of two types, standing or static. A standing query acts as an agent, constantly being evaluated to ensure that the collection of data represented by the document 110 is up-to-date relative to the query specification. As changes are made to the corresponding data source 112, the document 110 is updated and those updates are propagated to any viewer that may be displaying the document 110. A static query represents a snapshot of the data in the corresponding data source 112 at the time that the query was invoked. The document 110 representing the corresponding data source 112 is not updated when the data source 112 changes but may be

manipulated by other software agents or individuals interacting with the document 110 directly.

Once created, one or more documents 110 may be placed into a conference 102 by a participant 104 (e.g., by dragging a document 110 and dropping it into a conference 102), then opened and acted upon by various client applications, such as display/processing tools (e.g., map viewers, list viewers, analytical packages, etc.). Within each conference 102, the domain data (i.e., the data from the corresponding data sources 112 represented in the documents 110) is extended through the addition of visualization and control properties such as, for example, an associated color and/or symbol for displaying the data or an indication of what data has been selected by a participant 104 using a client application. The visualization and control properties become part of the data represented in the documents 110 allowing the client applications to focus on the presentation of the information rather than needing complex logic for accessing the data or logic dealing with collaboration between the participants 104 to a conference 102. Documents 110 may be graphically overlaid or textually combined to show relationships between data from different data sources 112 and to extract information that could not be extracted by viewing the data separately. Documents 110 can be attached to tasks and may be passed from place-to-place or person-to-person following a process.

Referring now to FIG. 5, the computer implemented collaboration system 10 of the present invention provides for single user collaboration. Single user collaboration is a concept used to describe a single user interacting with multiple visualization or data processing tools against one or more documents 110 within the collaborative context 100. By having all domain, control, and visualization properties available through the collaboration system 10, collaborative tools work together to extract information from the data and cooperate for problem solving. It is important to note that in accordance with the present invention, there is no direct communication between the tools.

Referring now to FIG. 6, the computer implemented collaboration system 10 of the present invention also provides for multi-user collaboration. Multi-user collaboration is an extension of the single user collaboration environment to include multiple participants 104. The collaborative framework of the collaboration system 10 provides inherent multi-user collaboration in that no specialized logic is required for client applications to act collaboratively. Multiple users enter conferences to

combine and apply various human knowledge, agent/application processing, and data resources to solve a problem. The computer implemented collaboration system 10 of the present invention permits collaboration between multiple users without requiring that images be pasted onto a common "whiteboard" in order for the multiple users collaborate on the same data. Instead, collaboration is accomplished directly within the tools. Additionally, collaboration between multiple users is possible without requiring the incorporation of special logic within the tools. It will be appreciated that in addition to human collaborators, there may be software agents involved in the collaborative process.

Referring now to FIG. 9, a block diagram of the components of one embodiment of a collaboration system 10 in accordance with the present invention is shown. The collaboration system 10 includes one or more repository servers 12, one or more data channel servers 14, a library server 16, one or more client data viewing tools 18 (e.g., a list viewer tool, a map viewer tool, or an X-Y viewer tool), a query viewer tool 20, and a conference manager tool 22. Each repository server 12 is enabled for accessing data within a corresponding data source 112, using data access methods native to its corresponding data source 112. It will be appreciated that since the repository servers 12 provide access to the data sources 112, the client data viewing tools 18 do not need to be enabled for accessing the data within the data sources 112, and therefore require no specific knowledge of the nature of the data within the data sources 112. The data channel servers 14 manage data centric channels within which extended data properties (e.g., visualization and control properties) are maintained. Maintaining the extended properties of the data within the data channel servers 14 rather than within the client data viewing tools 18 allows for single user and multiple user collaboration without requiring that the client data viewing tools 18 to be enabled for direct communication with one another or have any knowledge of each other.

The collaboration system may include additional management components supplied by the MITRE Corporation as part of the Joint Collaborative Services (JCS) Project, such as a JCS participant server 24, a JCS context server 26, and a JCS document server 28. The participant server 24 maintains a listing of all authorized participants 104 as well as the processing state of the participants 104 and the conferences 102 that they have entered. The document server 28 provides interfaces to manipulate documents 110 within folders. Interfaces provide for creation and

deletion of documents 110 as well as folder management to allow organization of documents 110 in a hierarchical storage structure. The context server 26 provides the interfaces to manage collaboration contexts 100 and conferences 102 within those contexts 100. The collaboration system 10 may also include such standard CORBA services as a naming service 30, a factory finder service 32 and a system service activation daemon 34.

Referring to Fig. 10, the components of the collaboration system 10 are organized into an N-tier infrastructure including a data management tier 50, an information access tier 52, a services tier 54, and a user interface tier 56. Each tier is made up of components accessed and manipulated through a defined interface. The infrastructure of the collaboration system 10 rides upon a CORBA communications framework. The data management tier 50 includes and the data sources 112 (e.g., a cities database, an airborne database). The data management tier 50 provides the data management capabilities normally supplied by database management systems.

The repository tier 52 is comprised of the repository servers 12 (e.g., a signal repository, a cities repository, an airborne repository, an airborne signal repository). The repository tier 52 provides the adaptive services to make the data maintained within the data sources 112 available to the services in the services tier 54 and the client tools in the user interface tier 56. Each repository server 12 in the repository tier 52 interacts with its associated data source 112 using the data source's 112 native access methods. This allows virtually any data source 112 to be integrated with the infrastructure without requiring modifications to the rest of the infrastructure services or client tools. The repository servers 12 in the repository tier 52 perform two functions. They act as proxies to execute service requests using their associated data source's 112 native access methods, and they provide requested data to the infrastructure in self-describing structures.

Requests are made to the repository servers 12 in two ways: standing queries and static queries. Upon initialization, each repository server 12 interrogates its associated data source 112 to extract the structure of the data maintained within it.

This definition is described as a feature type. Each repository server 12 then registers with the library server 16 providing the supported feature type and the type of queries that the repository can perform (blank, standing, static). When a query is executed, the result of the query is transformed in to a self-describing data structure made accessible through a component called a "feature collection."

The repository servers 12 are responsible for accepting requests for information, executing those requests and then managing the resulting collection of information. The collection of information resulting from a query, called a "feature collection," is made available in a self-describing format. The information and the access methods to manipulate the collection are modeled after the "Simple Features Specification" developed by the Open GIS Consortium. FIG. 35 illustrates the components that make up a "feature collection".

Each feature in a "feature collection" is managed in the form of a directed acyclic graph (DAG). The DAG structure is used to describe the information resulting from a query and is subsequently used to communicate (pass-by-value) the object information between the client and server. The DAG structure, which is illustrated in FIGS. 36-37 has three parts: (1) an array of properties that contain only attribute information; (2) an array of nodes that contain lists of attributes (element node) or lists of other nodes (node list); and (3) an array of edges that connect two nodes. It will be appreciated that the DAG Structure is easily converted from/to the DOM Objects.

The services tier 54 is comprised of the data channel servers 14, the library server 16, the participant server 24, the context server 26, and the document server 28, as well as other services. The services tier 54 provides services that are accessible to any other service, client tool or repository. The services tier 54 maintains the majority of the business logic as applied to a specific domain problem. The services tier 54 is designed to be extended, allowing domain specific business logic to be added and made available to the enterprise system. New services register their existence with the naming service 30 providing their home interface such that client tools and other services can learn and utilize their capabilities.

The user interface tier 56 is comprised of thin client applications/applets/servlets (the client tools 18) that allow the user to interact with the data. Each client tool 18 interfaces directly with the collection (if no collaboration is desired) or directly with the data channel(s) 14 (provides collaboration features).

Referring to FIGS. 11-13, the collaboration system 10 of the present invention provides an infrastructure for integrating legacy system capabilities and those provided by the collaboration system 10. The infrastructure of the collaboration system 10 provides a foundation for keeping up with rapidly changing technology and supports adaptation of new capabilities as systems evolve. The collaboration system

10 has an open architecture providing multiple options for integrating legacy systems. The level of integration selected for each legacy component depends on the capabilities of the infrastructure being utilized and the plans for system expansion. If long-term migration plans include extensive use of legacy software components, higher levels of integration are required to fully utilize the benefits of the architecture. If the plan is to make temporary use of legacy components until other capabilities are developed, a lower level of integration may be appropriate. One recommended approach provides for three levels of integration. This approach allows each component (data source, processing components, user interface) of the legacy system to be integrated as necessary to achieve the desired system capabilities.

Fig. 11 illustrates first (or minimum) level integration of the collaboration system 10 with a legacy system. First level integration requires no change to the legacy system. A repository 12 in the information access tier 52 is developed to provide access to the legacy data source 200. This level of integration allows access and manipulation of domain data by the existing tools 18 provided by the collaboration system 10 infrastructure. It allows full access to query and create documents from new and legacy data sources and allows existing viewing tools 18 (those provided with the collaboration system 10 infrastructure) to act on the data collaboratively without requiring changes to the legacy application 202 software.

Fig. 12 illustrates second (or midlevel) level integration of the collaboration system 10 with a legacy system. Second level integration involves modifying one or more of the legacy client viewers and/or processes to access the legacy data 200 through the collaboration system 10 infrastructure. In addition to having a new repository server 14 in the information access tier 52 associated with the legacy data source 200, the legacy application 202 is connected through a native languages API to the services tier 54. This enables selected portions of legacy applications 202 (combined user interfaces and processing applications) to operate in a collaborative environment and to manipulate the legacy data source 200 as well as all other data sources 112 made available to the infrastructure, while still maintaining the ability to interact directly with the legacy data source 200 using the legacy application 202.

FIG. 13 illustrates third (or full) level integration of the collaboration system 10 with a legacy system. Third level integration involves rewriting components (data viewers, processing) of the legacy system using the underlying component architecture of the collaboration system 10. This provides the benefits of component

distribution, system management, viewers that are Web enabled, and supports lifecycle management (activation, passivation, and persistence). As with first and second level integration, a new repository 14 is provided in the information access tier 52 that is associated with the legacy data source 200. However, in the case of full integration, the legacy application is rewritten as one or more thin viewers 204 included in the user interface tier 56 and a legacy processing service 206 included in the services tier 54. The thin viewers 204 may, for example, be rewritten in Java making them Web enabled and machine independent. Incorporating the legacy user interface and processing services into the user interface tier 56 and services tier 54, respectively, makes them a system component available for enterprise usage. It will be appreciated that, full integration of the collaboration system 10 with a legacy system lowers system maintenance costs, eliminates duplicate functionality across the enterprise, and makes each enhancement available to the entire enterprise. In addition, the integration technique chosen, and corresponding benefits, are managed stepwise with respect to both cost and risk, in accordance with project needs using the present invention.

Referring to FIG. 14, the collaboration system 10 of the present invention moves the complexity of collaborative processing into the infrastructure. Visualization and control properties (color, selection, symbology, etc.) become an extended part of the data within the infrastructure rather than simply being a hard-coded characteristic of the client viewing tool 18. In this approach, client applications (user interfaces, processing agents) are simplified by removing the need for specialized data access methods or collaboration implementation logic. Viewing tools 18 simply access the data through the infrastructure, display or manipulate the data as appropriate to the tool 18, and provide any updates back to the infrastructure. Any interactions with the data, including manipulating visualization characteristics, are viewed collaboratively by all tools 18 interacting within the same conference 102. Because all of the visualization and control properties are treated as an extension of the domain data, the infrastructure provides a natural environment for software agent technology to be applied as "collaborative agents" working to solve a problem. Agents can monitor and act on actions performed by human participants or can be configured to perform actions based on control information.

Referring now to FIGS. 15-22, exemplary user interfaces of the collaboration system 10 and several components thereof are shown. FIGS. 15-16 show an

exemplary embodiment of a user interface 60 of the collaboration system 10. The collaboration system user interface 60 may be configured to run within another application, such as a Web browser, or as a separate application within the operating system environment of the user terminal 106. The collaboration system user interface 60 provides for ease of access to the conferences 102 and information within a conference 102. In this regard, the various conference rooms 102 within a context 100 may be displayed in a left hand side panel of the collaboration system user interface 60. Windows associated with the various client tools 18 are displayed within a right hand side panel of the collaboration system user interface 60. The collaboration system user interface 60 allows multiple saved workspaces consisting of conferences 102 and tools 18. It also allows for the dragging and dropping of documents 110 into the various viewing tools 18. Additionally, the collaboration system user interface 60 permits easy navigation between conferences 102. There may be multiple active conferences 102 containing documents 110, participants 104, tools 18. Within a conference 102, a participant or group of participants 104 analyze information and interact to solve problems.

FIG. 17 illustrates interfaces of the query viewing tool 20 and view into JCS document server 28. The query viewing tool 18 dynamically learns about the repositories 12 and gets attribute metadata from the repositories 12. It creates an agent representing the standing query. The results of the query become a document 110 which may then be used for collaboration. The document itself is a token representing the results - no document data is conveyed to the user's viewer by this action. The documents 110 created by the standing query agents are displayed within the JCS document server 28 interface.

FIG. 18 shows an interface of the map viewing tool 18. The map viewing tool 18 may comprise an open source component such as, for example, the BBNOpen Map Viewer, which supports layering and a standards-based interface. The map viewing tool 18 displays a map in a chosen projections (e.g., a Mercator projection as is shown) with the data items overlaid on the map and colored in accordance with their extended properties in the data model. The map viewing tool 18 includes a configurable pop-up "layers editor" menu where the user may edit visualization attributes (e.g., line type, line color, fill color) for display of the data items.

FIG. 19 shows an interface of an extended properties editor 36. The extended properties editor 36 provides for attachment of extended properties (e.g., color,

highlight, visibility, label, symbol, etc.) to data items in the data channel(s) 14. The extended properties editor 36 dynamically learns the information schema from the repositories 12. The rules applied through the extended properties editor 36 run as agents within the data channel(s) 14.

5 Fig. 20 shows an interface of the X-Y data viewing tool 18. The X-Y data viewing tool 18 allows the users to select X and Y attributes from the list provided by the repositories 12 for display within one or more plots, provides for reordering of the plots, and permits zooming and panning in any plot independently or independently.

10 FIG. 21 shows an interface of the list viewer tool 18. The interface of the list viewer tool 18 provides for viewing and manipulation of data items from the data sources 112 in a table format. In this regard, the data items may be sorted. Specific rows within the data table may be selected, colored, and hidden. Additionally, the participants 104 may select various attributes of the data items for viewing.

15 Fig. 22 shows an interface of the chat tool 18. The chat tool 18 supports multi-user conversations from multiple conferences 102 in multiple contexts 100. As a shown by the example text within the chat tool 18 interface, participants 104 connect to a document 110 and communicate with one another. Participants 104 in the same conference 102 see the same visualization properties such as color and visibility of participant inputs.

20 It will be appreciated that the previously described collaboration system 10 infrastructure provides a change to the way systems are built and enhanced. Using the collaboration system 10 infrastructure, new capabilities can be added to the system as small client applications that interact through the infrastructure. The resulting system is constructed of many small applications providing unique capabilities that work
25 together to form the entire system. Each client user interface, processing component, or data repository interacts in a data centric collaborative environment where each component capability extends the capabilities of the other components. The result is a system whose overall capability grows exponentially with every added capability. With the collaboration system 10 infrastructure, each user is free to select the
30 appropriate tools 18 to be most effective at analyzing and manipulating data no matter what the data source 112. This allows human resources with varying backgrounds (engineering, analytical, mathematical, operational, etc.) to use specialized tools that enable the most effective application of their diverse skills to solve problems. In this regard, the performance metrics of one embodiment of a computer implemented

collaboration system 10 in accordance with the present invention are summarized in FIG. 23.

Referring now to FIGS. 24-27, the collaboration system 10 of the present invention provides information access services (also referred to as library services).

5 The information access services (IAS) are composed of a set of factory components, management components, library components, and request components that provide methods for discovery of available data sources 112 and the creation of requests for information from those data sources 112. These components are based on the United States Imagery and Geospatial Services (USIGS) Geospatial and Imagery Access
10 Services (GIAS) Specification. FIG. 24 illustrates the high level interaction between libraries, managers and requests. FIGS. 25-27 illustrate the lower level interaction between the (IAS) components in performing a query on a data source 112 and subsequently retrieving data to be used by a client tool 18.

Features of the various interface components in FIGS. 24-27 are summarized
15 in the table below.

20	• Library:	“Named” Object within the production domain that supports information access capabilities. All IAS services accessible through the Library Object. Database location, data representation (schema, object model), and type (RDBMS, OODBMS, file) are transparent to users of IAS.
25	• Standing Query Manager:	Is responsible for initiating the client request and then managing the request objects over the duration of the transactions. Other types of Managers (Query Manager, Agent Manager, etc.) support different forms of information access.
30	• Standing Query Request:	Client query transactions result in the creation of a Request Object. The Request provides the client visibility into the information access process. The client has three methods of being notified when information is available: Post a callback for a-synchronous notification; Synchronously block until information is available; or Poll for Request status periodically.
35		
40	• Feature:	Provides a common adapter (interface) to a domain object. Through the Feature, a client can access a domain object’s information. The Feature and Repository Objects provide an adapter layer that

shield client programs from the difference database storage and access mechanisms.

- 5 • Property: A Directed Acyclic Graph (DAG) of Properties is used to retrieve/update the information on a particular Domain Object. Associations between two Features (Facility -> Equipment) is represented as a DAG property that contains a sequence of feature association structures. From this property (within Facility) a client can create a second collection of Features (Equipment) that can be displayed.
- 10
- 15 • Feature Type: The SIGINT Object Model (SOM) and Fusion Object Model (FOM) have identified a set of core classes (Features - Installation, Facility, Equipment, Unit, Signal, etc) that make up a domain. Through the Feature Type a client can obtain retrieve meta-data that is used to construct a query.
- 20
- 25 • Property Def: A Directed Acyclic Graph of Property Def's (DAG Def.) is used to pass the definition of a particular Domain Object.
- 30
- 35 • Repository: Provides a common interface to a storage server for query evaluation and management. Each Feature Type within a database will have an associated Repository Object. The Query Request created by the Query Manager goes to the Repository for evaluation. The Repository is responsible for converting the query, which is in the domain terms, into the specific language and schema of the database. The Repository performs the query and populates the Feature Collection with feature objects.
- 40 • Factory: Provides services for construction of instance objects. There is a specific factory for each class. Multiple construction methods may be provided depending on the factory.

The inheritance structure of the various IAS components is illustrated in FIGS. 28-32.

Referring more particularly to FIG. 25 when a user of the collaboration system 10 activates the query viewer client tool 20, the query viewer client tool 20 dynamically learns about repository features via the library server 16. In this regard, the query viewer client tool 20 gets the query manager from the library server 16, which includes a library and a feature types manager. The feature types manager in

turn accesses a feature type within the repository server 12. The feature type includes a property definition and an installation definition. Using the query viewer client tool 20, a query is submitted via the library server 16 to the repository server 12. In this regard, when the repository server 12 receives a query request created through the library server 16, the repository server 12 creates a standing query request. The repository server 12 then creates a document 110 (also referred to herein as a feature collection) and also executes the query. The standing query request is executed through a repository and a data store to access data within a data source 112 associated with the repository server 12. The repository server 12 creates a feature for each domain data item meeting the specified query criteria. Each feature created includes an extended property. The document 110 created in response to the query is returned in the form of a directed a-cyclic graph (DAG) to the query viewer client tool 20.

Referring now to FIGS. 33-34, the data channel is the collaborative interface to the data provided by a document 110. A data channel server 14 is created when a document 110 is placed into a conference 102. Upon initialization, the data channel server 14 is extended to provide visualization and control properties such as highlight, visibility, and color. The data channel server 14 is extendable from client applications or agents in real time by calling methods on the extended properties manager to teach the data model additional collaborative attributes. FIG. 33 shows the data channel services framework in relation to other component interfaces within the collaboration system 10 architecture.

^{Sub A3} FIG. 34 illustrates the components that make up a data channel server 14 and describes the interactions between a client and the data channel sever 14 to learn about the data referenced by a document 110 and to extract the information through the data channel server 14 interface, as well as register for updates that the data channel server 14 may receive. As is shown in FIG. 34, the data channel server 14 includes a conference 102. Within each conference 102 there are multiple data channels. Each data channel includes a data model. Each data model represents multiple data items having multiple extended properties. Each data model maintains the current version of each of its data items. When a client data viewing tool 18 is started, the desktop manager provides a handle to the a viewer within the client data viewing tool 18. The viewer includes a view which includes an item presentation. The view maintains the most recently received version of the data model obtained by

the client data viewing tool 18 from the data channel server 14. In this regard, the client data viewing tool 18 gets the data model from the data channel server 14 and registers with the data channel server 14 to be informed of events that the data channel receives from the data model. The next step undertaken by the client data viewing tool 18 is to get the DAG definition of the properties of each data item. In this regard, the client data viewing tool 18 asks the data channel server 14 for only the information needed for rendering its display. Next the client data viewing tool 18 gets all of the changes to the data model. Then, as events are received, the client data viewing tool 18 asks for any updates to the data model since the last version of the data model was obtained from the data channel server 14.

Several features of the present invention are applied to reduce a required network bandwidth for collaboration and to reduce data copying across the network. These mechanisms avoid some known performance problems with distributed object systems.

First, the repository sets policies to access the data it manages. This allows "lazy evaluation" of queries, postponing actual querying until the data is needed. The repository also has control of how many queries are supported, the ability to bundle updates, and the ability to limit the amount of data retrieved in a collection. Typically, the repository is placed topologically and computationally close to the data source to minimize network usage between the data source in the repository.

The feature collection is implemented as a CORBA proxy, that is, a token, so that no matter how many users and conferences the data is represented in, the collection itself is created and managed exactly once. The feature collection may be located topologically and computationally near the repository, where creation and updates of collections minimize network communications bandwidth and latency.

The data channel is selected via a "finder" service, which has the ability to find the best data channel manager for the particular collection and conference. The data channel uses two mechanisms to optimize its performance vis-a-vis the viewers: first, viewers receive only the features that they request, and secondly, the data changes are not sent to all subscribers immediately. Instead, version change events are sent, which viewers can manage in the best way suited to their behavioral use (e.g., ignoring events altogether, responding to at most one event every 10 seconds, displaying the availability of an update but requiring a user to take action to receive the update).

While various embodiments of the present invention have been described in detail, further modifications and adaptations of the invention may occur to those skilled in the art. However, it is to be expressly understood that such modifications and adaptations are within the spirit and scope of the present invention.